IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.: 09/553,971
Filed: April 21, 2000
Inventors:
Sai V. Allavarpu
Rajeev Angal
Anand Bhalerao

Title: Thread-Safe Portable Management Interface

§

Examiner: Shah, Nilesh R.
Group/Art Unit: 2195
Atty. Dkt. No: 5181-48600

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below.

Robert C. Kowert
Printed Name

Signature

December 19, 2005
Date

PRE-APPEAL BRIEF REQUEST FOR REVIEW

Mail Stop AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

Applicant requests review of the final rejection in the above-identified application. No amendments are being filed with this request. This request is being filed with a notice of appeal. The review is requested for the reason(s) stated below. Claims 1-38 are pending in the application. Reconsideration of the present case is earnestly requested in light of the following remarks. Please note that for brevity, only the primary arguments directed to the independent claims are presented, and that additional arguments, e.g., directed to the subject matter of the dependent claims, will be presented if and when the case proceeds to Appeal. Applicants note the following clear errors in the Examiner's rejections.

The Examiner rejected claims 1-38 under 35 U.S.C. § 112, second paragraph, as being indefinite. Specifically, the Examiner asserts that claims 1, 15 and 27 are unclear at to which requests are being sent to the primary scheduler. Applicants traverse this rejection. Claim 1 clearly refers to a secondary scheduler executable to receive a plurality of requests from a multi-threaded application in a thread-safe manner and send the requests to the primary scheduler in a thread-safe manner. Thus, claim 1 is not indefinite as the Examiner contends, but instead clearly denotes which requests are being sent to the primary scheduler. Similarly, both claims 15 and 27 recite, in part, "receiving a plurality of management requests", "scheduling the plurality of management requests ... after receiving the management requests", "sending the management requests" and "executing the management requests". Thus, both claim 15 and claim 27 are quite clear regarding which requests are being sent. The Examiner also argues that claims 6 and 7 are unclear as to which requests are being sent. Regarding claims 6 and 7, the Examiner contends, "it is unclear which requests are being sent, (i.e. requests from the primary or secondary scheduler?)" (parentheses by Examiner). Claim 6 makes no reference to the sending of any requests, but instead recites the limitation "wherein the requests comprise management requests". Claim 1 refers to a secondary scheduler executable to receive requests *from a multi-threaded application* and to send the requests *to*

the primary scheduler. Thus, applicants assert that the language of claim 6 is quite clear regarding requests. Similar remarks as those above regarding claim 6 also apply to the Examiner's objection to claim 7 as well.

The Examiner also rejected claims 1-38 under 35 U.S.C. § 103(a) as being unpatentable over Kimmel et al. (U.S. Patent 6,105,053) (hereinafter "Kimmel") in view of Maresco (U.S. Patent 6,418,458). Applicants traverse this rejection and submit that **the Examiner has failed to provide a *prima facie* rejection of independent claims 1, 15 and 27.**

Contrary to the Examiner's assertion, Kimmel in view of Maresco fails to teach or suggest a primary scheduler that is executable to schedule requests for networked data resources. Kimmel teaches an operating system for non-uniform memory access (NUMA) multiprocessor systems and utilizes a software abstraction of the system hardware to maintain balanced processor and memory loads (Kimmel, Abstract). The Examiner cites column 24, lines 14-26 and column 6, lines 43-65 of Kimmel. The Examiner's first cited passage (column 24, lines 14-26) describes how Kimmel's operating system may function in systems having various numbers of execution queue levels and that have differently sized thread groups. The second passage cited by the Examiner describes the thread group structure maintaining cumulative timeslice and job processor (JP) accounting for all threads in a thread group. This cited passage also describes how a JP dispatcher selects a thread group to execute. Kimmel teaches that a dispatcher selects an individual thread from a thread group based on the local priority and scheduling policy. Selection occurs at two levels. Global scheduling policies are used to select a thread group, while local scheduling policies are used to select an individual thread from the selected thread group.

Neither of the Examiner's cited passages teaches or suggests a primary scheduler executable to schedule *requests for networked data resources*. Kimmel's JP dispatcher selects a thread to execute on a processor of the NUMA system. Kimmel fails to mention any networked data resources. **All resources taught by Kimmel are local to a single NUMA system and are not networked data resources.** Additionally, Kimmel makes no mention of *requests* for networked data resources. Kimmel states that the "dispatcher is a kernel subsystem that is a mechanism responsible for scheduling and executing processes on an associated JP [job processor] in accordance with certain global and local scheduling policies" (Kimmel, column 5, lines 43-46). Kimmel is concerned with balanced processor and memory loads in a NUMA system. Kimmel clearly does not pertain to *scheduling requests for networked data resources*.

Maresco pertains to the creation of threads to prioritize the execution of tasks in an object-oriented system (Maresco, column 1, lines 42-57; column 2, lines 25-38). **Neither Kimmel nor Maresco has anything to do with scheduling requests for networked data resources.** All resources taught by Kimmel are local to a single NUMA system and are not networked data resources. And the threads in Maresco are clearly not requests for networked data resources. Thus, Kimmel and Maresco clearly do not teach or suggest anything about scheduling requests for networked data resources.

In the Response to Arguments, the Examiner reasserts that Kimmel "teaches request [*sic*] for networked data requests", citing the abstract, column 1, lines 30-55, column 3, lines 1-15, column 4, lines 39-65, column 24, lines 14-26, and column 6, lines 43-65 of Kimmel. The Examiner fails to respond to the previous arguments concerning the cited portions of columns 24 and 6. The cited portion of column 1 is from Kimmel's background section describing a *separate* system, which "discloses a method for affining groups of related threads from the same process to a group of JPs to improve secondary cache affinity while improving efficiency of operations among threads in the same group and reducing overhead for operations between groups." The cited passage further discloses the "need for an operating system having a global scheduling mechanism that may be implemented in a scalable multiprocessor system having a NUMA architecture." Contrary to the Examiner's assertion, Kimmel, at column 1, lines 30-55, does not mention anything about requests for networked data resources. As stated above, Kimmel fails to mention any networked data resources. All resources taught by Kimmel are local to a single NUMA system and are not networked data resources.

Additionally, Kimmel makes no mention of *requests* for networked data resources. Kimmel is concerned with balanced processor and memory loads in a NUMA system. Kimmel is not concerned with *scheduling requests for networked data resources*. In the portion of column 3 cited by the Examiner, Kimmel describes that "each run queue that is associated with one of the remaining nodes identif[ies] the active process affined to groups of the processors that share the resource associated with the node." As stated shown above, this relates to a single multi-processor system and does not describe *scheduling requests for networked data resources*. The portion of column 4 cited by the Examiner refers to JP caches, and, as shown above, is also irrelevant to *scheduling requests for networked data resources*. Likewise, Kimmel's abstract is completely devoid of any mention of scheduling requests for networked data resources.

Similarly, the Examiner's cited portions of columns 24 and 6 also fail to mention anything at all about requests for networked data requests." Instead, column 6, lines 43-65 describe Kimmel's method for the JP's dispatcher to select a thread group to execute and column 24, lines 14-26 is merely a statement by Kimmel that his invention may be implemented in forms other than those specifically described by Kimmel. As with the Examiner's other cited passages, neither the cited portion of column 6 or of column 24 mention anything about requests for networked data resources.

Additionally, Kimmel in view of Maresco also fails to teach or suggest a secondary scheduler that is executable to receive a plurality of requests from a multi-threaded application in a thread-safe manner and send the requests to the primary scheduler in a thread-safe manner. The Examiner cites several passages of Kimmel referring to Kimmel's medium term scheduler. However, Kimmel's medium term scheduler does not receive any requests from a multi-threaded application. Instead, Kimmel's medium term scheduler "monitors the progress of active processes in the system and sets a flag for those processes that are not progressing" (Kimmel, column 2, lines 40-45). **Nowhere does Kimmel describe his medium term scheduler receiving requests from a multi-threaded application.** Furthermore, the medium term scheduler in Kimmel's system does not send requests received from a multi-threaded application to Kimmel's dispatcher, which the Examiner equates to the primary scheduler of claim 1. The Examiner has not pointed out any aspect or feature of Kimmel's system that can be interpreted as a secondary scheduler executable to *receive requests from a multi-threaded application* and to *send the requests to a primary scheduler*.

Instead, Kimmel's medium term scheduler monitors thread groups to identify languishing thread groups and candidates for load balancing by monitoring the loads of the respective scheduling locals (i.e. certain nodes in Kimmel's hierarchical representation of a system's hardware) to identify any load imbalances and by identifying any thread groups that do not have the same current and home scheduling locales (Kimmel, column 10, lines 34-40). As described at one of Examiner's cited passages (Kimmel, column 10, lines 50-65), when Kimmel's medium term scheduler identifies a languishing thread group, it may do one of three things. First it may raise the thread group's priority. Secondly it may assign the thread group to the run queue of a higher node. And thirdly the medium term scheduler may set a flag associated with the thread group to allow poaching of the thread group by another JP. Thus, Kimmel's medium term scheduler has absolutely nothing to do with receiving requests from a multi-threaded application. Nor does Kimmel's medium term scheduler send the requests to the JP dispatcher, which the Examiner equates to a primary scheduler. Instead, as noted above, the medium term schedule monitors the state of various thread groups through Kimmel's thread group structure and changes priorities and scheduling locales accordingly to maintain balanced processor and memory loads.

In the Response to Arguments, the Examiner cites column 2, lines 30-45, column 3 lines 5-15, column 5, lines 43-58, column 6, lines 49-61, and again cites column 8, lines 24-32, column 9, lines 9-37, column 10, lines 50-65, and column 12, lines 56-65, without any response to, or rebuttal of, Applicants' previously presented arguments, outlined above. The Examiner merely asserts, "Kimmel teaches a secondary scheduler is executable to receive a plurality of requests from a multi-threaded application and send the requests to the primary scheduler." However, none of the passages cited by the Examiner actually describe any sort of secondary

scheduler that receives requests from a multi-threaded application and sends them to a primary scheduler. Instead, the cited portion of column 2 discloses a medium term scheduler, which, as shown above, does not receive requests from a multi-threaded application and thus cannot be considered the secondary scheduler of Applicants' claim. The cited portion of column 3 describes that each active process or thread-group has scheduling locales, e.g., home-scheduling locale and a current scheduling locale. As shown above, scheduling locales are not relevant to a secondary scheduler executable to *receive requests from a multi-threaded application* and to *send the requests to a primary scheduler.* Applicants have already shown this in previous office action responses and above, and the Examiner has failed to accurately rebut any of these arguments. The cited portions of columns 5 and 6 also fail to support the Examiner's contention. Instead, the cited portion of column 5 describes Kimmel's dispatcher and medium term scheduler while the cited portion of column 6 refers to Kimmel's dispatcher and global and local scheduling. As shown above, the dispatcher of Kimmel's system can not be considered the primary scheduler of claim 1 since Kimmel's dispatcher it does not schedule requests for network data resources sent from a secondary scheduler.

Maresco is relied upon by the Examiner only to teach "a thread safe system" and the Examiner cites column 2, lines 30-38. However, Maresco fails to overcome any of the above noted deficiencies in Kimmel regarding claim 1. Thus, the Examiner's combination of Kimmel in view of Maresco would not result in a system that includes a primary scheduler which is executable to schedule requests for networked data resources; and a secondary scheduler, wherein the secondary scheduler is executable to receive a plurality of requests from a multi-threaded application in a thread-safe manner and send the requests to the primary scheduler in a thread-safe manner. Instead, even if the references were properly combinable (which Applicants do not concede), the Examiner's proposed combination of Kimmel and Maresco would at most result in Kimmel's system for maintaining balanced processor and memory loads in a NUMA system where Kimmel's thread groups are the work crews of Maresco. Such a combination has no relevance to Applicants' claim 1.

Moreover, in regard to claims 1 and 15, Maresco does not teach or suggest anything about receiving or sending requests for networked data resources in a thread-safe manner. Instead, Maresco is concerned with safely *creating* threads for task execution is a computer system (Maresco, column 1, lines 42-57; column 2, lines 25-38). Maresco teaches nothing about a scheduler receiving and sending requests in a thread-safe manner. Just because Maresco mentions thread creation for task execution does not imply that Maresco suggests receiving and sending requests by schedulers in a thread-safe manner. As discussed above, neither Kimmel nor Maresco has anything to do with scheduling requests from a multi-threaded application, let alone receiving and sending requests for scheduling in a thread-safe manner. Applicants note that the Examiner has not responded to the arguments above relating to Maresco.

Additionally, in regard to claim 15, Kimmel in view of Maresco fails to teach or suggest scheduling the plurality of management requests in a secondary queue in the secondary scheduler after receiving the management requests from the manager application. The Examiner cites several passages of Kimmel referring to Kimmel's medium term scheduler. However, as discussed above, Kimmel's medium term scheduler does not receive any requests from a multi-threaded application and clearly does not receive management requests from a multi-threaded manager application. Instead, as noted above, Kimmel's medium term scheduler "monitors the progress of active processes in the system and sets a flag for those processes that are not progressing" (Kimmel, column 2, lines 40-45). Furthermore, Kimmel's medium term scheduler does not schedule management requests, but instead monitors and adjusts the priority of threads in thread groups executing on a multi-processor system.

In further regard of claim 15, Kimmel in view of Maresco also fails to teach or suggest sending the management requests from the secondary scheduler to a primary scheduler in a thread-safe manner. as discussed above, Kimmel's medium term scheduler does not receive any requests from a multi-threaded application and Maresco does not teach sending such requests in a thread-safe manner.

Furthermore, in regard to claim 15, Kimmel in view of Maresco fails to teach or suggest executing the management requests on the managed objects after scheduling the management requests in the primary queue. The Examiner cites column 6, lines 62-67 where Kimmel describes how using thread groups in developing processes give a user flexibility "to choose between creating a new thread within an existing thread group or creating a new thread group." The cited passage makes no mention of executing management requests on managed objects. Creating thread groups and giving users the ability to choose between adding a new thread to an existing thread group or creating a new thread group has no relevance to management requests or managed objects, as they are understood in the art. As noted above, nowhere does Kimmel mention anything regarding either management requests for managed objects. Maresco is only relied upon the by the Examiner to teach a thread safe system. Maresco fails to overcome any of the above noted deficiencies of Kimmel.

Similar remarks as those above regarding claim 15 also apply to claim 27.

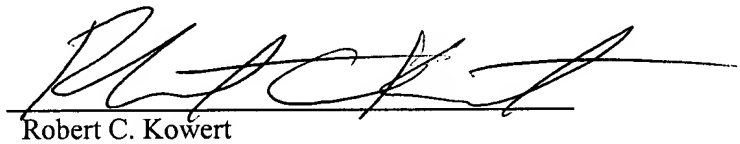
For at least the reasons above, the current rejections are clearly erroneous and removal thereof is respectfully requested.

In light of the foregoing remarks, Applicant submits the application is in condition for allowance, and notice to that effect is respectfully requested. If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such an extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 501505/5181-48600/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard
☒ Notice of Appeal

Respectfully submitted,



Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850
Date: December 19, 2005